The Report committee for Carlos Townes Gillett
certifies that this is the approved version of the following report:

A Comparison of Two Markov Chain Monte Carlo Methods for Sampling from Unnormalized Discrete Distributions

APPROVED BY
SUPERVISING COMMITTEE

Stephen Walker, Supervisor

James Scott

A Comparison of Two Markov Chain Monte Carlo Methods for Sampling from Unnormalized Discrete Distributions

by

Carlos Townes Gillett, B.A.

Report

Presented to the Faculty of the Graduate School
of the University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science in Statistics

The University of Texas at Austin
May 2015

A Comparison of Two Markov Chain Monte Carlo Methods for Sampling from Unnormalized Discrete Distributions

by

Carlos Townes Gillett, MSStat

The University of Texas at Austin, 2015

SUPERVISOR: Stephen Walker

This report compares the convergence behavior of the Metropolis-Hastings and an alternative Markov Chain Monte Carlo sampling algorithm targeting unnormalized, discrete distributions with countably infinite sample spaces. The two methods are compared through a simulation study in which each is used to generate samples from a known distribution. We find that the alternative sampler generates increasingly independent samples as the scale parameter is increased, in contrast to the Metropolis-Hastings. These results suggest that, regardless of the target distribution, our alternative algorithm can generate Markov chains with less autocorrelation than even an optimally

scaled Metropolis-Hastings algorithm. We conclude that this alternative algorithm represents a valuable addition to extant Markov Chain Monte Carlo Methods.

Contents

List of Figures

1. Introduction

The purpose of this report is to compare the convergence behavior of two Markov Chain Monte Carlo (MCMC) algorithms. One, the Metropolis-Hastings, is one of the most popular of such methods (Cowles & Carlin, 1996), while our proposed alternative algorithm is new and relatively untested (Walker, 2014). The goal is to explore through simulation study how the two algorithms perform under different circumstances and in which instances, if any, the alternative appears to be preferable to the Metropolis-Hastings. We find that the alternative algorithm displays consistently decreasing autocorrelation as the scale parameter increases. This is more easily exploited when the target distribution is highly monotone, but holds for the less monotone distributions simulated here and can theoretically be extended to most if not all unnormalized discrete distributions over infinite space. The Metropolis-Hastings, in contrast, appears to have a finite set of optimal scale parameters for a given target distribution that minimize the autocorrelation. The ability of the alternative to produce significantly less correlated samples need to be explored more formally, but could make it a valuable addition to extant MCMC techniques.

We begin by providing a brief introduction to Bayesian inference and MCMC to provide the necessary context to appreciate this study. Both algorithms are then described in detail in section 2. Section 3 char-

acterizes the unnormalized Poisson distribution used in the simulation study. Sections 4 and 5 each describe a component of the simulation study and present its findings. The first component of the simulation study involves generating multiple, independent Markov chains from each algorithm. The second compares the two algorithms' performance through the generation of long, dependent chains. In section 6 we discuss our findings in aggregate and offer a theoretical justification and outline future research, and section 7 concludes the report.

Markov Chain Monte Carlo methods have in large part been developed in order to solve problems arising from Bayesian statistical inference (Gelfand & Smith, 1990). For this reason, we will provide a brief introduction to this topic before explaining what MCMC is and how it is used. We note here that this introduction to Bayesian statistics is at best cursory. The information presented here can be found in Bernardo and Smith (1994) and this source should be consulted for further reading.

Bayesian inference is a class of statistical reasoning in which the parameters of a statistical model, traditionally considered fixed, unknown constants, are treated instead as random variables themselves. Bayes' rule is then applied in order to obtain the probability distribution of the parameters conditioned on the observed data. Recall that Bayes' rule refers to the following statistical identity about conditional probabilities

$$p(A|B) = \frac{p(B|A)p(A)}{P(B)}$$

Consider a generic statistical problem in which we have observed data y we believe to be distributed according to some probability distribution based on parameter θ. Here θ is unknown and we wish to estimate it based on the data observed. We can consider the likelihood function, written $L(y|\theta)$ as the probability of observing our data given a particular value of θ. In traditional, frequentist statistics often the value of θ that would maximize this likelihood is then taken as the estimate of the unknown parameter.

To the Bayesian, however, this θ is itself a random variable which is believed–before looking at the data–to be distributed according to some probability distribution $\pi(\theta)$. This initial $\pi(\theta)$ is referred to as the prior distribution, or simply the prior. This prior should be constructed based on previous experimentation and or expert opinion and such that the support of θ corresponds to the range of possible values of the parameter in our probability model.

Bayes' rule can be applied to obtain what is referred to in Bayesian statistics as the posterior distribution, here called $\pi_p(\theta|y)$. The posterior then reflects our updated belief about the distribution of the parameter of interest θ after having seen our data. We now have

$$\pi_p(\theta|y) = \frac{L(y|\theta)\pi(\theta)}{p(y)}$$

The posterior distribution now represents a combination of our preconceived notions (the prior) with the information revealed to us by the data (the likelihood). Because the marginal distribution in the denominator, $p(y)$, will be a constant and only contribute to the normalizing constant of our posterior distribution, the posterior is often left simply as

$$\pi_p(\theta|y) \propto L(y|\theta)\pi(\theta)$$

As long as our likelihood and prior are valid probability measures, this will result in a valid posterior distribution. However, depending on the distributions of the prior and likelihood, the posterior may not be easily integrated.

Often we are interested in functions integrated over the posterior distribution. For instance, it would be natural to use the mean of the posterior distribution as a point estimate of θ, or to want to know the variance of the posterior distribution.

If the posterior distribution has a form for which the integral is difficult or impossible to solve, we turn to numerical estimation. A very efficient way of estimating the integral of a probability distribution is

to approximate numerous samples from said distribution. There are numerous ways to do this, but MCMC is one of the most efficient and therefore widely used methods (Gamerman & Lopes, 2006).

Markov Chain Monte Carlo is an area of statistics concerned with simulating samples from a probability distribution that is difficult or impossible to sample from directly (Geyer, 1992). MCMC works through iterating a stochastic, or Monte Carlo, process to generate a Markov chain that converges to the desired, or target, distribution. We note here that this is at best a cursory introduction to MCMC and point to Gamerman and Lopes (2006) for further reading.

A Markov chain is defined as a sequence of random variables in which each variable in the sequence is dependent only on that variable immediately preceding it. This can be written mathematically as

$$p(\theta^{(t+1)}|\theta^{(0)}, \ldots, \theta^{(t)}) = p(\theta^{(t+1)}|\theta^{(t)}) \quad t = 1, \ldots, \infty$$

Here $\theta^{(t)}$ represents random variable θ at point t in the sequence. Let us also define here the state space, or support, S of θ, as the set of all possible values the random variable can take. Formally we have

$$\theta^{(t)} \in S \quad t = 1, \ldots, \infty$$

This conditional probability, $p(\theta^{(t+1)}|\theta^{(t)})$, defines the Markov chain.

Note that in a Markov chain the first component of the sequence, here $\theta^{(0)}$, needs to be drawn from some other probability distribution. Note as well that here and throughout this report we will consider Markov chains over a discrete time and state space, as this is both convenient for introduction and the most relevant to the analysis conducted.

The Markov chains used in MCMC have another property, namely that they possess a defined stationary, or limiting distribution. What this means is that as the sequence continues infinitely, the marginal distribution of the tth random variable approaches some probability distribution, here called $\pi(\cdot)$, that may or may not be equal to the Markov chain's transition probability (Gamerman & Lopes, 2006). Mathematically we can say that $\pi(\cdot)$ is the stationary distribution for the Markov chain defined by transition probability $p(\theta^{(t+1)}|\theta^{(t)})$ if

$$\sum_{\theta^{(t)} \in S} \pi(\theta^{(t)}) p(\theta^{(t+1)}|\theta^{(t)}) = \pi(\theta^{(t+1)})$$

If a Markov chain has a limiting distribution, as the number of iterations t approaches infinity, the marginal distribution of the tth random variable in the sequence will approach π regardless of the initial value of the sequence or what distribution it was generated from. Therefore as we continue the Markov chain for an increasing number of steps, the sequence of random variables will increasingly approximate, or converge to, random variables from the limiting distribution.

Certainly not all Markov chains have limiting distributions, and proving a given transition probability will converge to a target distribution can be difficult. In this paper we use three sufficient conditions to show a given chain has the desired limiting distribution, namely reversibility, positive recurrence, and aperiodicity (Tierney, 1994).

Reversibility, also referred to as the detailed balance condition, means that the probability of moving from state θ to state ϕ is equal to the probability of moving from state ϕ to state θ. A Markov chain is positive recurrent if it can move from any state to any other state in a finite number of steps with non-zero probability. A Markov chain is aperiodic if it has no period greater than 1, meaning that the chain can always return to its current state in the next step and that there is no set number of iterations greater than one for which there is zero probability of returning to its current state. These conditions will be shown to hold for the MCMC algorithms used in this report.

We can now see how Markov chains are used in Bayesian inference. When the integral of the posterior distribution does not have a closed form solution, we can simulate values of this distribution with a Markov chain to estimate the integral. This is done by constructing a transition probability we are able to sample from that has our desired distribution as its limiting distribution and sampling from that transition probability numerous times. After a sufficient number of iteration we have an

approximate sample from the desired posterior distribution. We have not yet, however, specified how quickly a Markov chain approaches its target distribution.

The question of how many Monte Carlo samples are required to be considered valid approximations of samples from the target distribution is of paramount importance in stochastic simulation (Cowles & Carlin, 1996). In general a researcher faces the tradeoff between wanting to have as many samples from the target distribution as possible on one hand and a finite amount of time and computing power on the other. The optimal number of samples taken then depends on the desired number of samples, the computational complexity of the sampling process, the time available to the researcher, and how quickly a given Markov chain converges to its stationary distribution. These factors will all vary, but in all cases a faster converging–also referred to as mixing–Markov chain is preferred.

This is the primary concern of this report; to determine which of two MCMC algorithms, the Metropolis-Hastings or our proposed alternative algorithm, converges to the target distribution in fewer iterations.

2. Background

2.1 Metropolis-Hastings

The Metropolis-Hastings algorithm, first described in papers by Metropolis et al. (1953) and Hastings (1970), is an MCMC method for generating samples from a distribution that is otherwise difficult to sample from. It is an iterative algorithm, drawing each new sample from a transition density conditioned on its current state. As such, it is by definition a Markov random process.

The popularity of the Metropolis-Hastings algorithm is due in part to it's wide range of applications (Gelman, Roberts, & Gilks, 1996). It can be used to sample both discrete and and continuous data, and works well in high dimensions (Hastings, 1970). Here we are only concerned with its use on discrete data, as the alternative proposed is only applicable to discrete distributions. Another positive feature of the Metropolis-Hastings is that the target probability distribution only needs to be known up to the normalizing constant.

The Metropolis-Hastings differs from other MCMC methods in that it's transition density is comprised of a proposal and acceptance probability. The algorithm works by generating proposals from some proposal density, and either accepting or rejecting them based on an acceptance probability.

Suppose we want to generate samples of random variable θ with support S and probability distribution characterized by $\pi(\theta)$. To simulate draws form this distribution with a Metropolis-Hastings algorithm, we first construct proposal density $q(\phi, \theta)$, that will generate a proposed new value, ϕ, given current state θ. Then define acceptance probability $\alpha(\phi, \theta)$ such that

$$\alpha(\phi, \theta) = \min\left\{1, \frac{\pi(\phi)q(\theta, \phi)}{\pi(\theta)q(\phi, \theta)}\right\}$$

We pause to note here why the target distribution need only be known up to the normalizing constant as $\pi(\cdot)$ appears in both the numerator and denominator of the acceptance probability. The algorithm then proceeds iteratively as follows:

1. Select arbitrary initial value $\theta^{(0)}$, initialize iteration counter (t).
2. Draw $\phi \sim q(\phi, \theta^{(t)})$
3. $\theta^{(t+1)} = \begin{cases} \phi \text{ with probability } \alpha(\phi, \theta^{(t)}) \\ \theta^{(t)} \text{ with probability } 1 - \alpha(\phi, \theta^{(t)}) \end{cases}$
4. Repeat from 2 until desired number of samples are acquired.

As stated in the introduction, Markov chain is valid and has limiting distribution $\pi(\cdot)$ if the transition probability has the qualities of reversibility, positive recurrence, and aperiodicity. We now show that these conditions hold for the Metropolis-Hastings described here.

The construction of the acceptance probability $\alpha(\phi, \theta)$ assures the reversibility of the Metropolis-Hastings. The Metropolis-Hastings satisfies the detailed balance condition if the following statement is true

$$\pi(\theta)q(\phi,\theta)\min\left\{1, \frac{\pi(\phi)q(\theta,\phi)}{\pi(\theta)q(\phi,\theta)}\right\} \;=\; \pi(\phi)q(\theta,\phi)\min\left\{1, \frac{\pi(\theta)q(\phi,\theta)}{\pi(\phi)q(\theta,\phi)}\right\}$$

Here the $\alpha(\cdot,\cdot)$ function is spelled out to demonstrate how the acceptance probability ensures the equality.

Now so long as $q(\phi, \theta)$ is constructed such that the chain is positive recurrent and aperiodic we know the Markov chain has a unique limiting distribution. The proposal function used for our Metropolis-Hastings sampler study is based a scale or spread parameter $k > 1$ defined as follows

- If $\theta - k + 1 \geq 1$, sample with equal probability from $\{\theta - k + 1, \theta - k + 2, \ldots, \theta + k - 2, \theta + k - 1\}$
- If $\theta - k + 1 < 0$, sample with equal probability from $\{1, 2, \ldots, \theta + k - 2, \theta + k - 1\}$

It is fairly trivial to show this proposal function makes the algorithm positive recurrent. For any current state $\theta \in S$, and any other state $\phi \in S$ there must be a finite distance between θ and ϕ. Further, the structure of the algorithm guarantees that there is always positive

probability of remaining in state θ and of moving at least one integer in the direction of ϕ. Therefore there is always positive probability of moving from any θ to any ϕ in S in a finite number of steps. Proof of aperiodicity is similarly trivial, as there is clearly no cyclical nature to the proposal distribution.

We can now formalize that the transition probability of the Metropolis-Hastings algorithm as

$$p_M(\theta^{(t+1)}|\theta^{(t)}) = q(\theta^{(t+1)},\theta^{(t)})\alpha(\theta^{(t+1)},\theta^{(t)}) + \boldsymbol{I}\left\{\theta^{(t+1)}=\theta^{(t)}\right\}\left(1-\sum_{\theta^{(t+1)}} q(\theta^{(t+1)},\theta^{(t)})\alpha(\theta^{(t+1)},\theta^{(t)})\right)$$

and are satisfied this transition has limiting distribution $\pi(\theta)$ and will approach this distribution as the number of samples approaches infinity.

One aspect of the Metropolis-Hastings worth discussion here is its sensitivity to scaling (Gelman, Roberts, & Gilks, 1996). In our application, this refers to how large a value of k we select for the proposal distribution. The algorithm will function for any k that is greater than 1, but different values of this scale parameter will affect how quickly the the Markov chain converges to the target distribution. Setting k too low limits our proposals to a small area around the current state. This results in a high acceptance rate, but slows the movement through the sample space. Setting k too large means there are a wide range of potentially proposed values around any current state. Theoretically there is some optimal scaling, or set of scalings,

that balance this tradeoff and provide the most efficient traverse of the sample space.

The optimal scale setting for a Metropolis-Hastings here would be the value of k which results in a Markov chain that converges to the target distribution in the fewest iterations. It is difficult to formally prove that one Markov chain is converging faster than another, making this hard to define. This topic has been the subject of a considerable body of generally work without a clear answer emerging. Practically we are left with the general rule that the optimal scaling is the one resulting in an acceptance rate of roughly 20% to 50%, with one-dimensional problems on the higher end of that spectrum (Gamerman & Lopes, 2006). We bring this up now as this characteristic of the Metropolis-Hastings is not present in the alternative algorithm, and we believe this difference can explain the difference in performance in the simulation study.

2.2 Alternative Algorithm

The alternative algorithm proposed here, introduced by Walker (2014), is another MCMC algorithm that can be used to sample from a discrete distribution for which the normalizing constants are unknown. Like the Metropolis-Hastings algorithm, it is a stochastic process that simulates samples from a target distribution through iterative sampling of a

transition probability.

Unlike the Metropolis-Hastings, the alternative sampler's use is limited to simulating discrete distributions with infinite sample space. Applying this alternative algorithm to a continuous state space, as will be seen below, would require the ability to integrate the target distribution. If we had this ability, we would no longer need this algorithm to simulate the target distribution. If the sample space of the target distribution were bounded, we would be able to calculate the normalizing constant and therefore would not need to use the alternative algorithm.

The alternative algorithm proposed here works differently from the Metropolis-Hastings in that it does not involve a proposal-acceptance routine. Rather, for a specified scale parameter $k > 1$ the algorithm constructs a local approximation of the target distribution of some length between k and $2k - 1$ around the current state θ and samples from the values over that range. That is, for a given k and current state θ, probability of moving to state ϕ is given by

$$p_A(\phi, \theta) = \frac{\pi(\phi)}{k} \sum_{i=\max\{\theta,\phi\}}^{\min\{\theta+k-1,\phi+k-1\}} \frac{1}{\sum_{j=\max\{1,i-k+1\}}^{i} \pi(j)}$$

where $\pi(\cdot)$ again represents the unnormalized target distribution. The alternative algorithm then proceeds iteratively as follows:

1. Set initial value of $\theta^{(0)}$, begin iteration count (t)

2. Calculate $p_A(\phi, \theta)$ for $\phi \in S_k$ where $S_k = \begin{cases} \{\theta-k+1,\ldots,\theta+k-1\} \text{ if } \theta-k+1 \geq 1 \\ \{1,\ldots,\theta+k-1\} \text{ if } \theta-k+1 < 1 \end{cases}$
3. Sample ϕ from S_k according to the probabilities calculated and take $\theta^{(t+1)} = \phi$
4. Repeat from 2 until the desired number of samples are acquired

The transition probability represents a valid probability measure over the range from $\{\max\{1, \theta - k + 1\}, \ldots, \theta + k - 1\}$ as it can be seen that:

$$\sum_{\phi=max\{1,\theta-k+1\}}^{\theta+k-1} p_A(\phi|\theta) = 1$$

As with the Metropolis-Hastings, we need to show that this alternative algorithm generates samples that will asymptotically approach the target distribution. Again we will do this by demonstrating its reversibility, positive recurrence, and aperiodicity.

The reversibility of this transition density is easily proven. Again, we have reversibility if

$$p_A(\theta, \phi)\pi(\phi) = p_A(\phi, \theta)\pi(\theta) \quad \forall \theta, \phi \in S$$

Inserting the relevant expressions gives us:

$$\frac{\pi(\theta)}{k} \sum_{i=\max\{\theta,\phi\}}^{\min\{\theta+k-1,\phi+k-1\}} \frac{1}{\sum_{j=\max\{1,i-k+1\}}^{i} \pi(j)} \pi(\phi) = \frac{\pi(\phi)}{k} \sum_{i=\max\{\phi,\theta\}}^{\min\{\phi+k-1,\theta+k-1\}} \frac{1}{\sum_{j=\max\{1,i-k+1\}}^{i} \pi(j)} \pi(\theta)$$

The bounds of the summations are identical on both sides of the equality [$\max\{\theta, \phi\} = \max\{\phi, \theta\}$, etc.] and the remaining terms are clearly equal.

The proof of positive recurrence is similar to that used for the Metropolis-Hastings. Again, for any current state $\theta \in S$, and any other state $\phi \in S$ there must be a finite distance between θ and ϕ. Further, the structure of the algorithm guarantees that there is always positive probability of remaining in state θ and of moving at least one integer in the direction of ϕ. Therefore there is always positive probability of moving from any θ to any ϕ in S in a finite number of steps, and by definition this alternative is positive recurrent. Proof of aperiodicity is similarly trivial, as there is clearly no cyclical nature to the proposal distribution.

The feature of this algorithm that most separates it functionally from the Metropolis-Hastings is how it responds to scaling. It is easy to see that as k approaches infinity, the transition probability distribution will approach the normalized target distribution. As discussed previously, the Metropolis-Hastings has some set of scale parameters which allow for the fastest mixing Markov chain. The alternative algorithm, however, will more closely approximate the target distribution at every iteration the larger the scale. There exists no theoretically optimal scale of the alternative algorithm, as with the Metropolis-Hastings. Rather,

increasing k should always yield a faster mixing Markov chain. This is reflected in our results, particularly in section 5, and expounded upon more in section 6.

Now that we have provided an introduction and justification of our two algorithms, we next describe the simulation study used to compare them.

3. Unnormalized Poisson

In our simulation study the target distribution is defined $\pi_N(\theta) \sim$ Poisson(λ), where λ represents the mean and variance characterizing a Poisson distribution. This distribution was selected for two reasons. First, it is a discrete distribution with a countably infinite state space, making it appropriate to simulate using the alternate algorithm. Second, depending on λ, the Poisson distribution can be more or less monotone. This allows for a richer comparison of the two algorithms over differently shaped target distributions.

Supposing we understood the weighting function of the Poisson distribution but not did not know it's normalizing constant, we define the unnormalized Poisson distribution

$$\pi(\theta) \propto \frac{\lambda^\theta}{\theta!} \; \theta \in \{0, 1, \ldots\}, \; \lambda > 0$$

and use this kernel for both simulation studies. We note here that all analysis was conducted using R statistical software (R Development Core Team, 2010).

4. Independent Chains

The first component of our study looks at the speeds at which both algorithms converge by looking at their behavior in numerous, short, independent chains. Simulating in this manner is outlined by Gelfand and Smith (1990). The idea is to generate N independent chains of length n from each algorithm. While each new sample in a chain is dependent on the previous one, the samples in two separate chains will be independent of one another, by the definition of a Markov chain.

Here we set the starting point to be the same for each chain, selecting a value relatively far from the true mean of the distribution. We then take the mean of all N samples at every iteration from $1, \ldots, n$. The idea is that after n iterations of the chain, the algorithms should have converged to be close to the target distribution. Therefore our sample of N independent draws should increasingly resemble the target distribution at each successive iteration. By setting both chains to have the same starting point for all N runs, the mean at iteration 1 will be exactly equal to the starting point for both chains. We can then plot the movement of the means from the initial value towards the mean of the target distribution. This allows for a visual inspection of the apparent speed at which each algorithms mean approaches the true mean.

This comparison is done at a 3 different levels of our spread parameter k, for 2 different levels of λ. In this way we can look at results for

both a more and less monotone target distribution, with narrow and wide proposal densities.

Figure 1 shows the results for a less monotone Poisson distribution with $\lambda = 10$ with the spread parameter, k, set to 3, 10, and 20. Each algorithm was run for 200 iterations 1000 times, and with all samples set to start at 0. With $k = 3$, the Metropolis-Hastings appears to converge at a faster rate, with its means hitting close to 10 several iterations ahead of the alternative algorithms. For $k = 10$ and $k = 20$, both algorithms approach the mean in only a few iterations and seem to get there at nearly the exact same pace.

Figure 2 shows the results generated when targeting a more monotone distribution. Here $\lambda = 2$, with k again set to 3, 10, and 20, respectively. Again we generated 1000 chains, this time only for 50 iterations each as they tended to converge more quickly, with all chains beginning at 9.

Here the alternative sampler performs better at for all values of k, especially as k increases. When $k = 3$, the difference appears marginal, although the alternative means are closer to 2 for all of the early iterations. But when k equals 10 and 20, the alternative means settle around 2 almost immediately. The Metropolis-Hastings performs best when $k = 10$, but it still takes approximately 10 more iterations to see means as close to 2 as the alternative.

The findings here suggest that for a less monotone distribution, or one with greater variance, the Metropolis-Hastings and alternative algorithm converge at nearly the same rate. The only possible exception is for small values of k, in which case the Metropolis-Hastings appears to converge faster. For the monotone distribution, however, we see the alternative algorithm out-performing the Metropolis-Hastings for all values of k tested here, and especially when k is large.

Figure 1: Means of 1000 Independent Chains for less Monotone Target Distribution

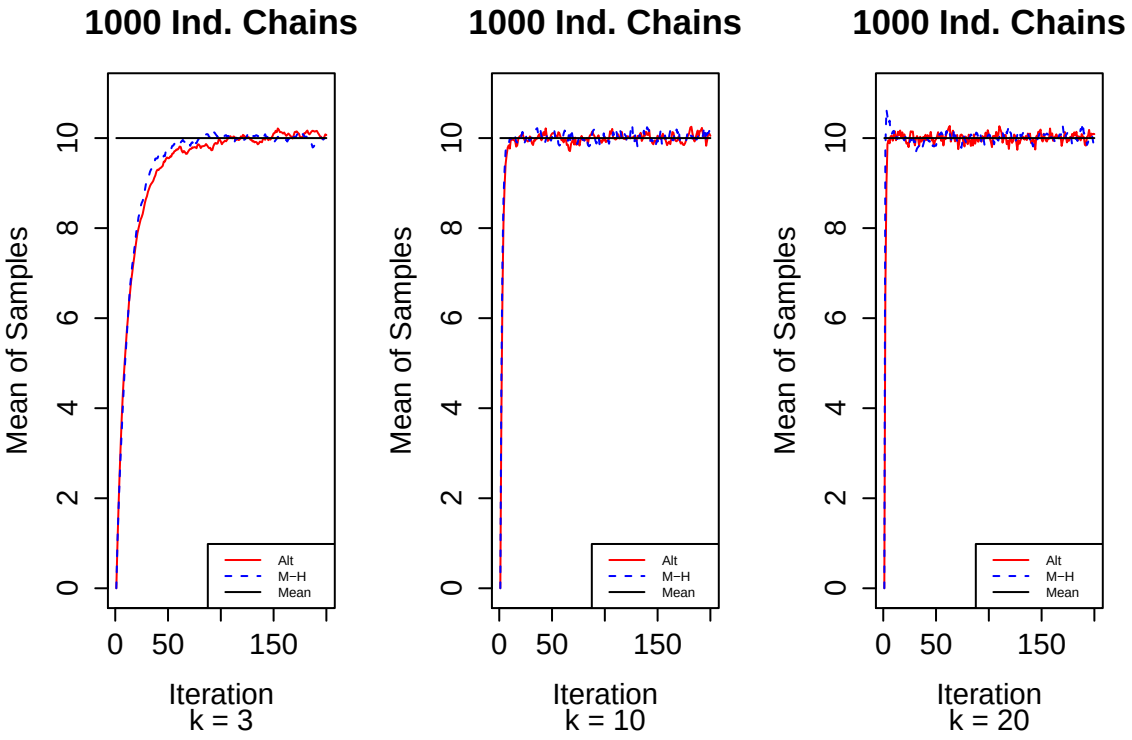
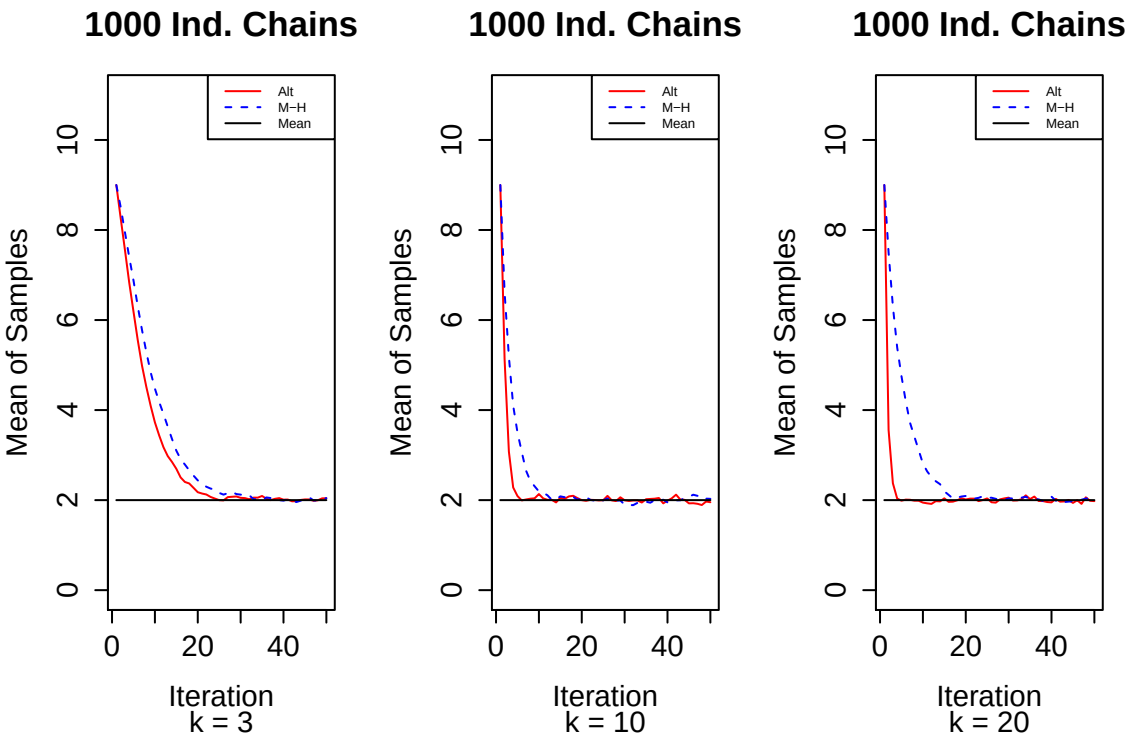


Figure 2: Means of 1000 Independent Chains for more Monotone Target Distribution



5. Dependent Chains

The second component of the simulation involves generating one long, dependent chain of from each algorithm. This is an approach proposed by Geyer (1992). We then look at the autocorrelation from each sample, and at trace plots to see the mixing and compare the autocorrelation between samples.

Trace plots are a basic visual inspection used to see how a Markov chain is moving around its sample space–also referred to as mixing. They are generated simply by plotting the Monte Carlo samples generated over their iteration, so the progress of the random process can be visualized. Trace plots are used, in part, to diagnose if a Metropolis-Hastings is scaled correctly. Too low a scale parameter and we will see the sampler taking very small steps and therefore many iterations to cover a meaningful range of the sample space. Too high a scale parameter and we can see the algorithm staying in one place for several iterations as it rejects many proposals.

The structure of a Monte Carlo sampler guarantees a certain amount of autocorrelation, but too high a degree of autocorrelation can be problematic as more iterations are necessary to provide a valid sample (Geyer, 1992). If a Markov chain is run long enough, the whole sample will eventually wash out the local autocorrelation. However, it is not always easy to run a sampler for long enough to assure the autocorrelation is not

an issue. Therefore in general the lower the degree of autocorrelation, the faster a Markov chain converges and the more efficient the algorithm. The autocorrelation function, for the lth lag, is defined (Box & Jenkins, 1976)

$$\rho_l = \frac{\sum_{i=1}^{N-l}(Y_i - \bar{Y})(Y_{i+l} - \bar{Y})}{\sum_{i=1}^{N}(Y_i - \bar{Y})^2}$$

and here is computed for every sample from lags $l = 0, 1, \ldots, 30$. The autocorrelation is another diagnostic of the scaling for the Metropolis-Hastings algorithm. Setting k too small or too large will result in higher autocorrelation. When scaled too small, the small step size produces high autocorrelation, while when the scale is too large high autocorrelation results from the many rejected proposals.

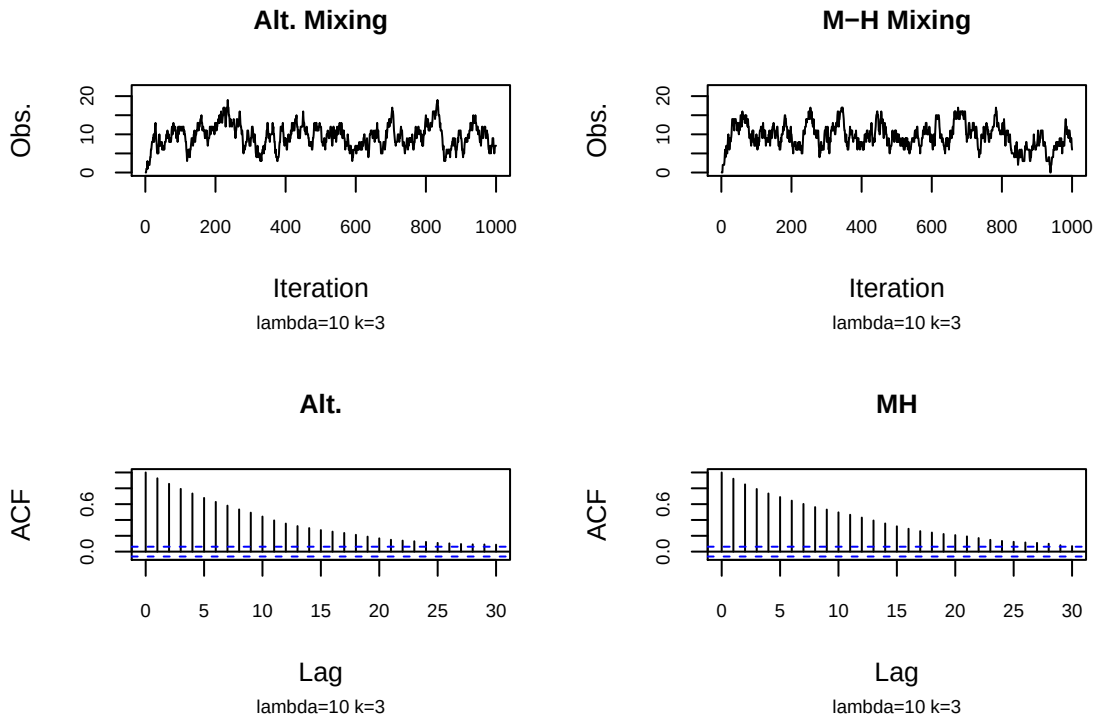
As in the first component of the simulation study, we compare the algorithms performance for both a less and more monotone target distribution at various levels of k. Each algorithm is run for 1000 iterations at each level setting. The results of these diagnostics appear in figures 3-5.

Figures 3-5 show trace plots and autocorrelation results for different values of k when targeting a Poisson with a mean of 10. When $k = 3$, both the trace plots and autocorrelations are very similar. Both trace plots show slow progress through the sample space, and the autocorrelations are high. This indicates the value of k is lower than

it should be for both algorithms. For $k = 10$ again we see very similar results between algorithms. Both show a lesser degree of autocorrelation than when k was three, and it is evident in the trace plots they are taking larger steps. The Metropolis-Hastings trace plot is slightly less dense than the alternative algorithms, as the larger k produces more rejections. Setting $k = 20$ appears to improve the mixing of the alternative sampler while worsening that of the Metropolis-Hastings. Here the alternative sampler shows almost no autocorrelation after the third lag, while the Metropolis-Hastings continues to show autocorrelation over 0.1 at least through lag 6. The trace plot of the Metropolis-Hastings shows it is frequently rejecting proposals, while the alternative sampler is moving much more frequently.

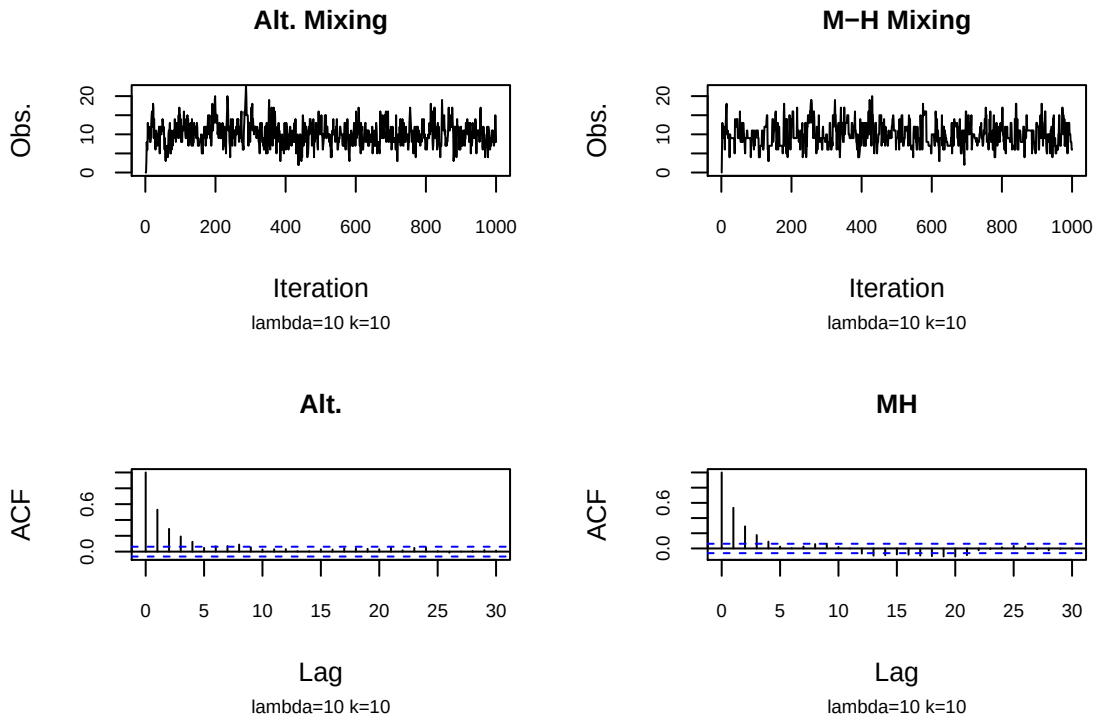
Figures 6-8 show the trace plots and autocorrelation results for the same three values of k this time targeting a Poisson with mean 2. Again both samplers show similar results for $k = 3$, with the alternative sampler showing a slightly higher degree of autocorrelation. With k set to 10, the alternative sampler shows a much lower degree of autocorrelation–nearly none after the third lag–and is clearly moving much more around the sample space. We can see the Metropolis-Hastings is rejecting a high percentage of its proposals, evidenced by its smoother trace plot. When $k = 20$ we see a similar but more extreme result, with the Metropolis-Hastings rejecting the majority of its proposals and demonstrating a much higher degree of autocorrelation than the alternative algorithm.

Figure 3: 1000 Iterations of Poisson with Mean 10 and Scale Parameter 3



We can see from these results that at larger values of k the alternative algorithm generates samples with lower autocorrelation, while the Metropolis-Hastings performed best in each situation when k was 10. However there is no evidence that $k = 10$ is the optimal scaling for the Metropolis-Hastings. To check that the alternative can produce samples with less autocorrelation than an optimally scaled Metropolis-Hastings, we calculated the acceptance rate for various values of k when targeting a Poisson with mean 10. For $k = 11$ the acceptance rate was just over 50%, decreasing to about 32% when $k = 20$. We then plotted the autocorrelation for each of those values and found it to bottom out when $k = 15$. We then performed the same comparison, this time with

Figure 4: 1000 Iterations of Poisson with Mean 10 and Scale Parameter 10



the alternative algorithm scaled to 30 and the Metropolis-Hastings to 15. The results are shown in figure 9. The Metropolis-Hastings displays marginally lower autocorrelation than when k was set to 10, but it is still clearly higher than that of the alternative. This is likely the lowest autocorrelation the Metropolis-Hastings can provide for this specific problem, while increasing k more for the alternative sampler should only further reduce the autocorrelation.

Figure 5: 1000 Iterations of Poisson with Mean 10 and Scale Parameter 20

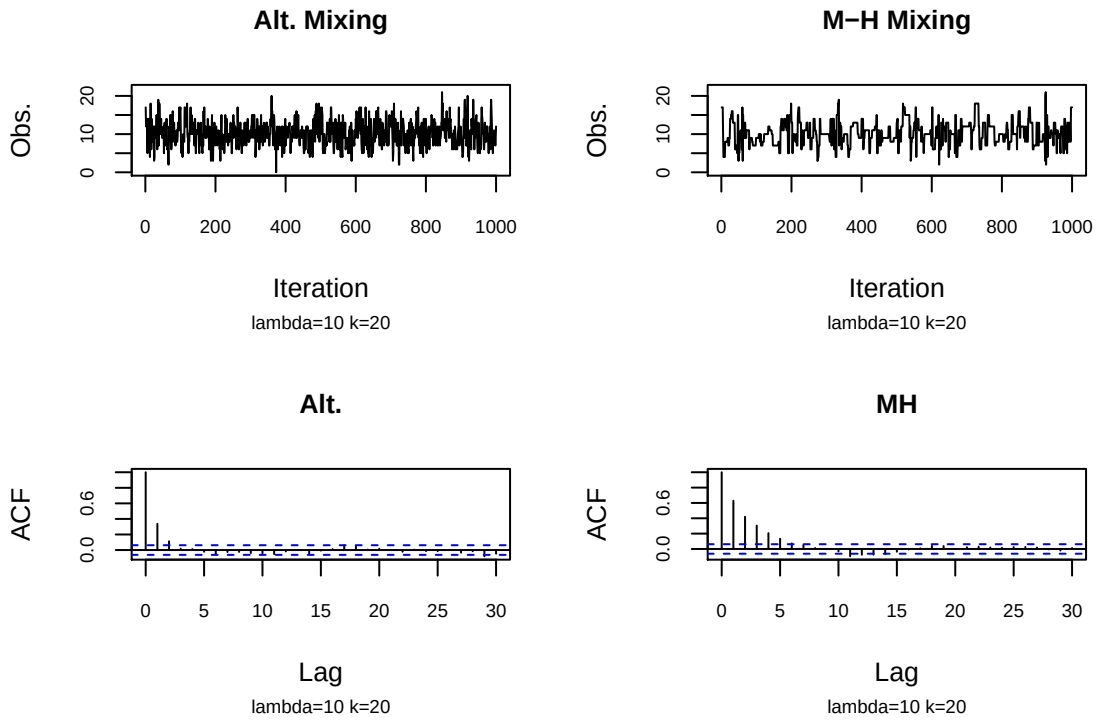


Figure 6: 1000 Iterations of Poisson with Mean 2 and Scale Parameter 3

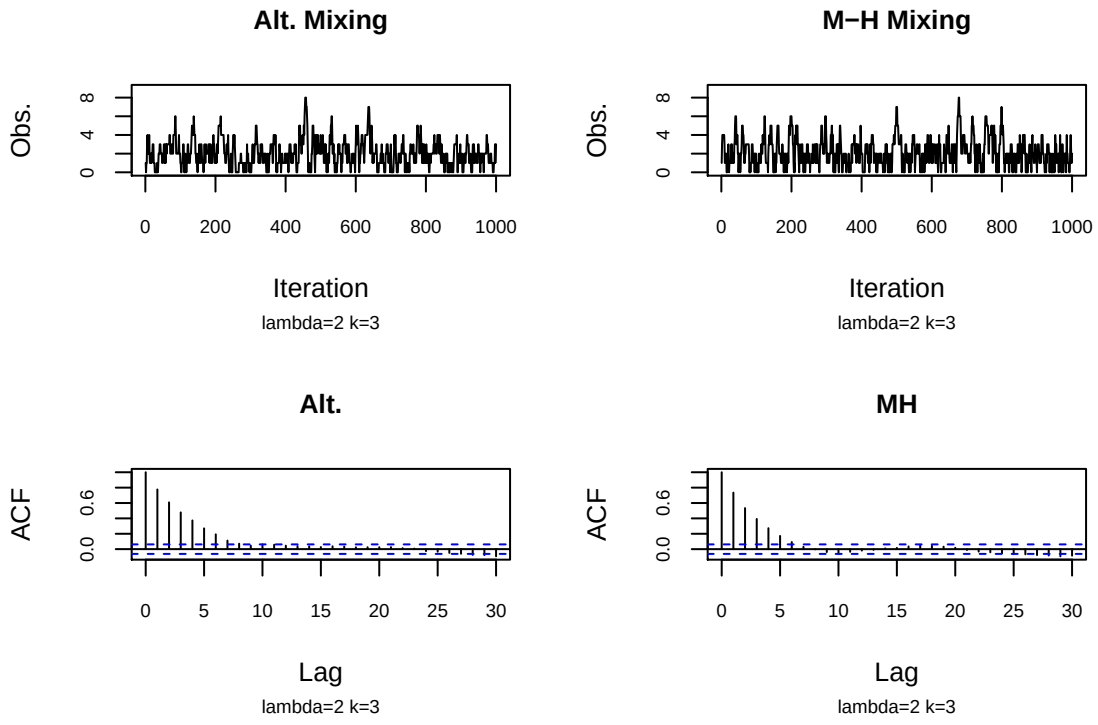


Figure 7: 1000 Iterations of Poisson with Mean 2 and Scale Parameter 10

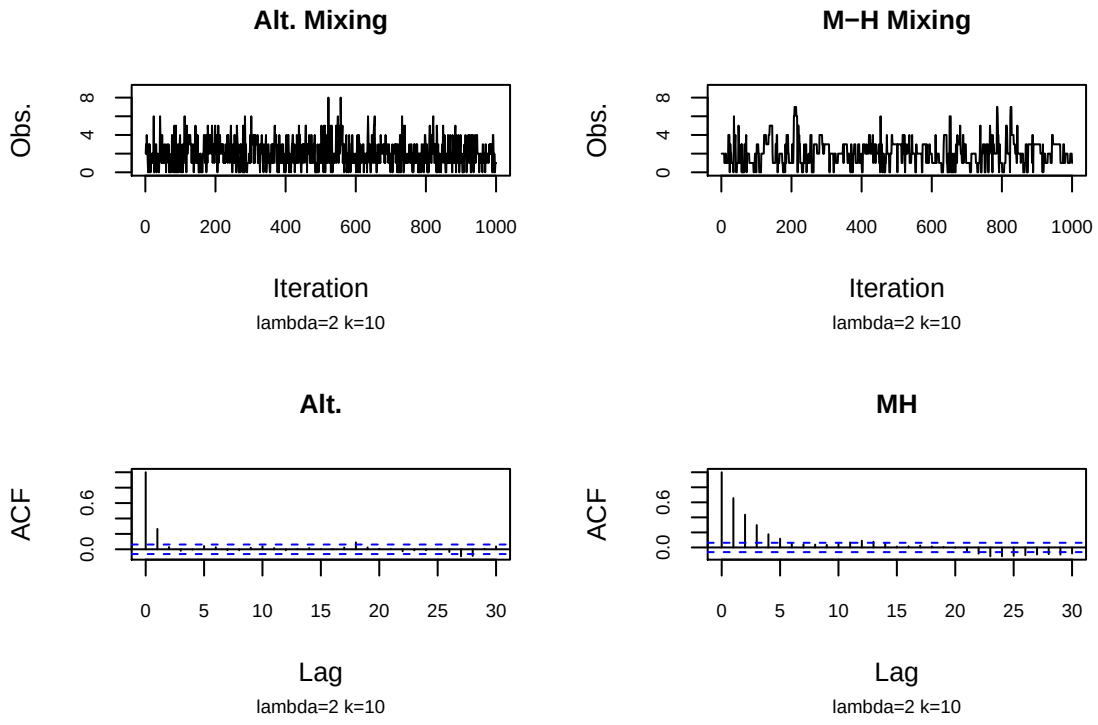


Figure 8: 1000 Iterations of Poisson with Mean 2 and Scale Parameter 20

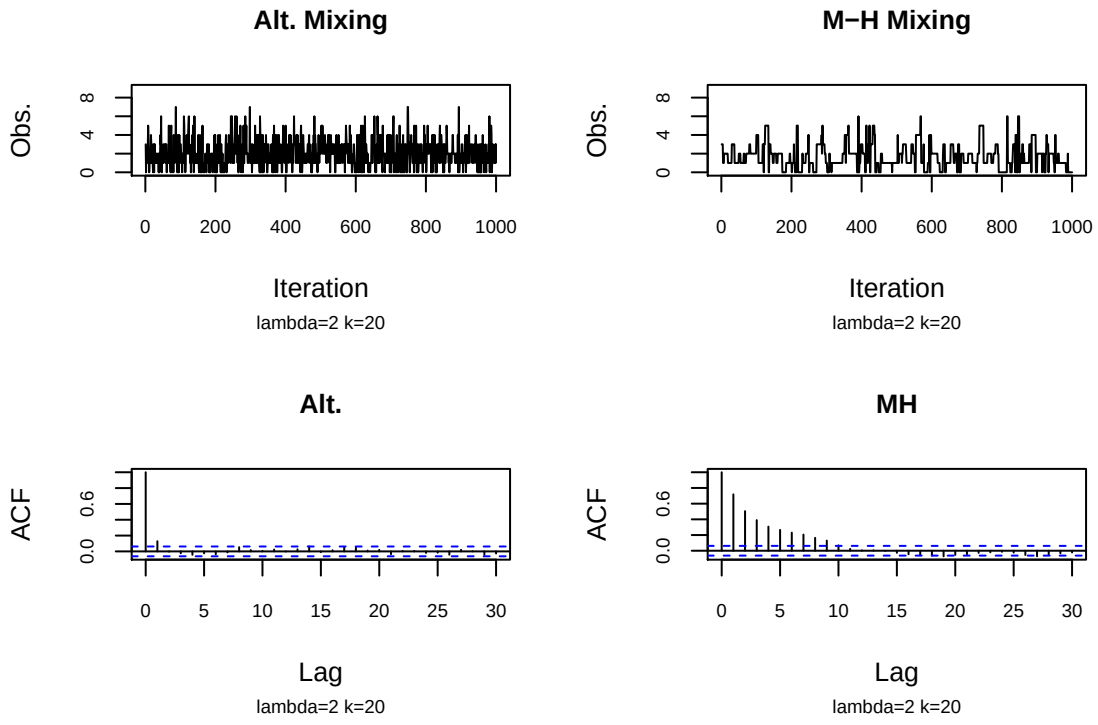
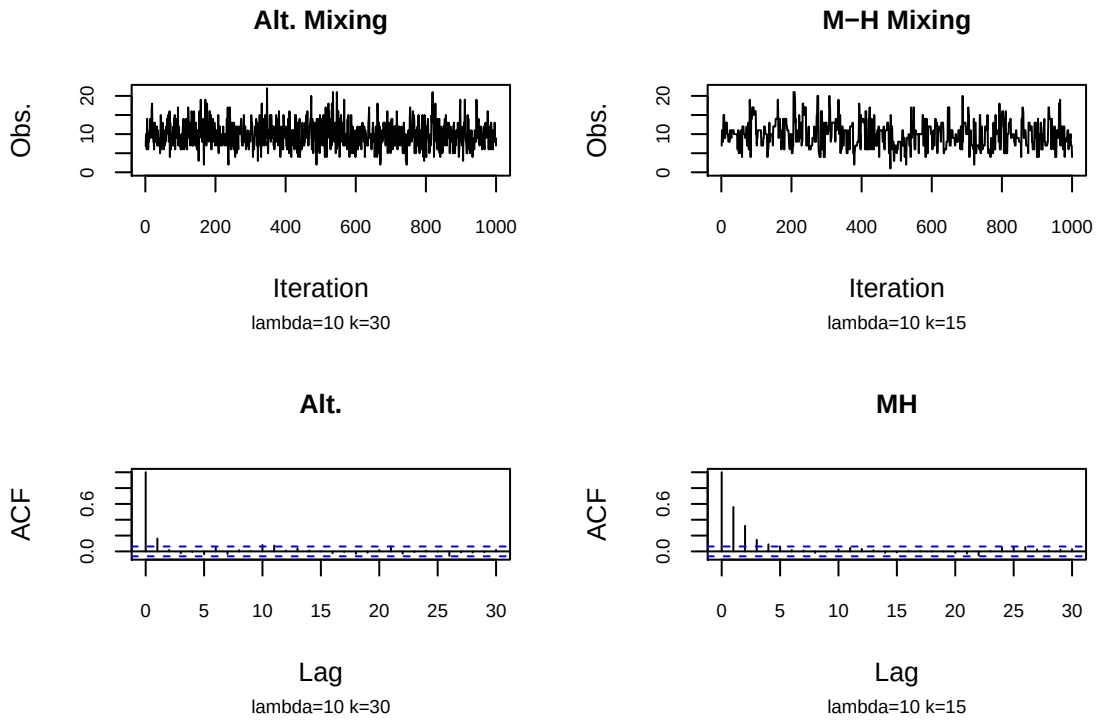


Figure 9: 1000 Iterations of Poisson with Mean 10 and Optimally Scaled Metropolis-Hastings



6. Discussion

6.1 Results

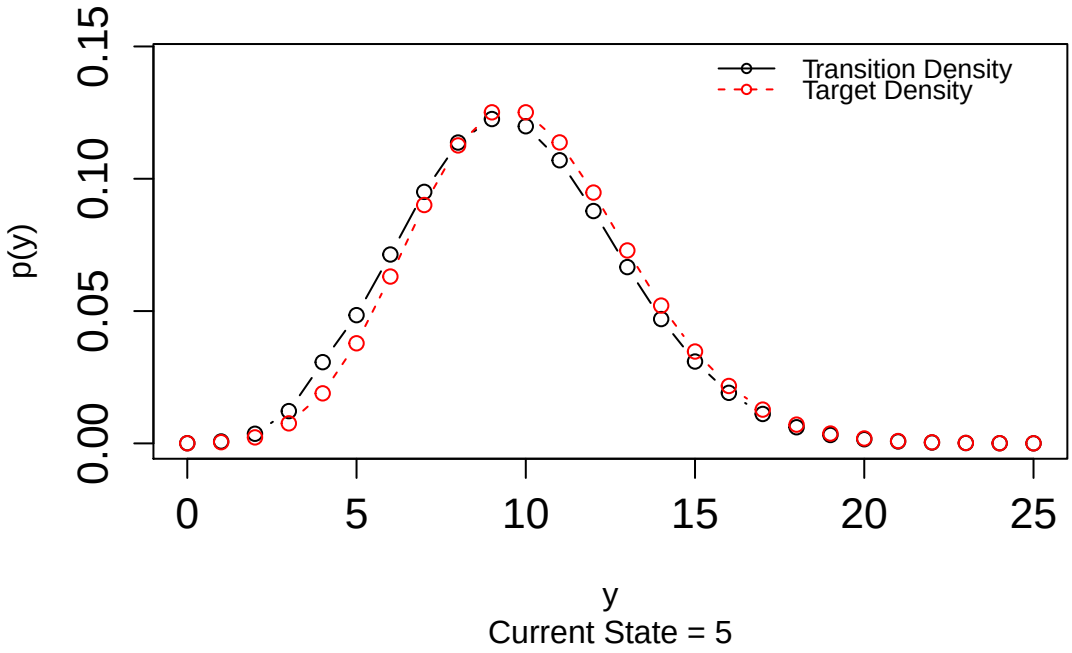
Our simulation studies suggest that the alternative sampler converges faster than the Metropolis-Hastings for highly monotone data, and that by increasing the spread parameter k can deliver samples with less autocorrelation. The question now is why this should be the case, and whether or not a formal argument can be made to generalize the result.

For any finite k, we are sampling from a subset of the total sample space S with probabilities proportional to those of the unnormalized target distribution $\pi(\cdot)$. We have stated earlier that as $k \to \infty$, $p_A(\phi|\theta) \to \pi_N(\phi)$, where again $\pi_N(\cdot)$ represents the normalized target distribution. Therefore for any distribution as k increases our transition probability will more closely resemble a draw form the target distribution.

We demonstrate this property in Figures 10-12, where the transition densities of the alternative algorithm are plotted alongside the true target density from three different starting points with a very large spread parameter. Here the mean of the target Poisson distribution is 10, and we plot the Markov transition density from current states of 5, 10, and 20. For all three plots $k = 100$. Note that here we only evaluate the density from 0 to 25 and omit the tails. From a current state of 10, the transition probability is nearly identical to the target density. From the

starting point of 20 the transition density deviates the most from the Poisson, but is still a very close approximation. These figures provide visual evidence of the transition probability's convergence to the target distribution.

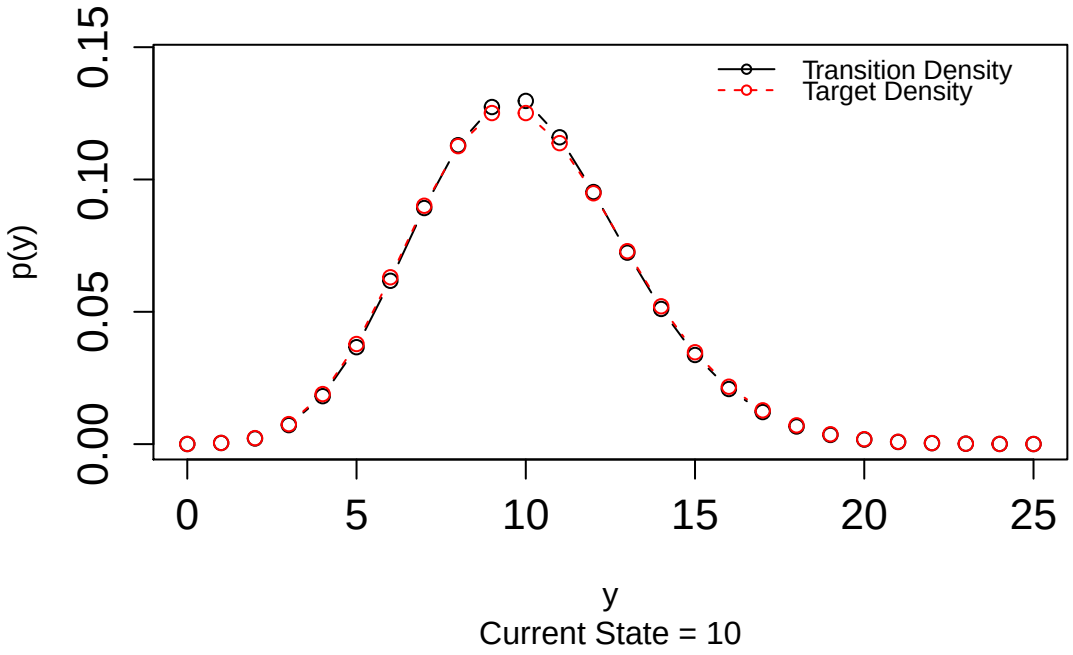
Figure 10: Transition Density from Current State $\theta = 5$ for Poisson with Mean 10 and Spread Parameter $k = 100$



As we will discuss further below, it is not always practical to increase k indefinitely. But for a fixed value of k, as our target distribution becomes increasingly monotone–for our Poisson example this would mean as λ approaches 0–a greater proportion of the probability mass will be between 0 and $\theta + k - 1$.

When in our simulation study we had a spread parameter of 20 for

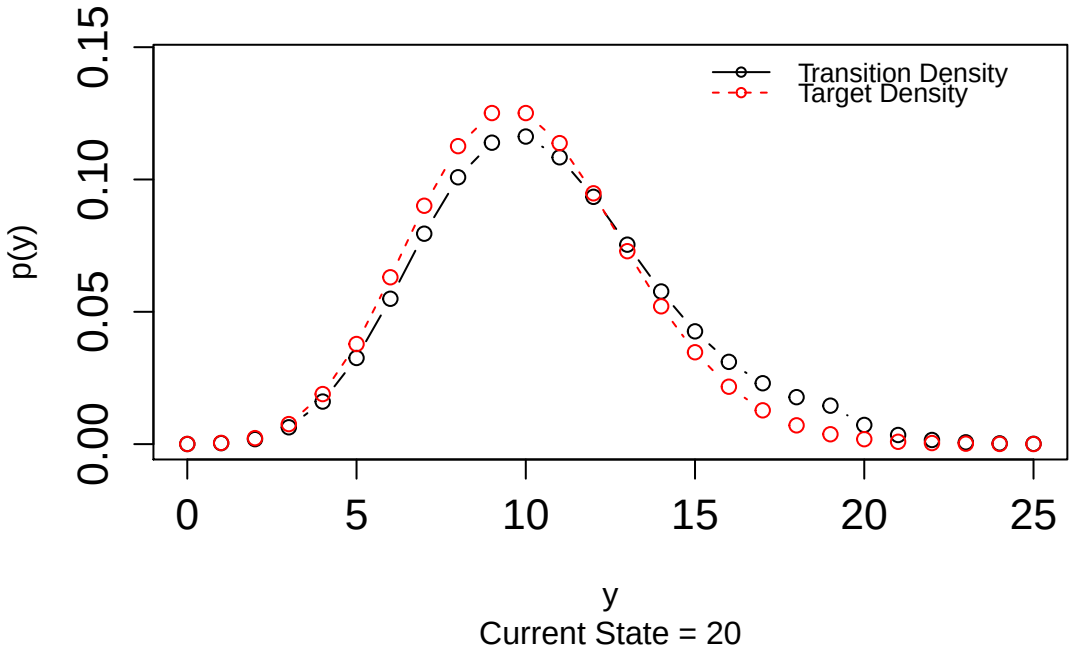
Figure 11: Transition Density from Current State $\theta = 10$ for Poisson with Mean 10 and Spread Parameter $k = 100$



a Poisson with mean 2, we were drawing from 0 to at least 20 (if the starting point was 0) with appropriately weighted probabilities. That minimal sampling range represents very close to 100% of the target distribution's probability mass, and therefore it is no surprise that this offers highly nearly independent samples that very closely resemble the target distribution.

While increasing k should always lead to lower autocorrelation, it may not always be practical to simply apply this alternate algorithm and set the spread to however high of a number is needed to get nearly independent samples. The tradeoff is that the higher k is, the greater

Figure 12: Transition Density from Current State $\theta = 20$ for Poisson with Mean 10 and Spread Parameter $k = 100$



the number of operations required of each iteration of the algorithm.

Therefore while it should always be possible to generate more independent samples with a larger spread, it may not be the best choice. For a widely dispersed distribution, an appropriately scaled Metropolis-Hastings with would likely be able to generate more samples in less time, negating the mild autocorrelative effects of the algorithm.

It is certain that the code used for the alternative algorithm can be improved upon, so it is possible that a more cleverly programmed routine could result in more instances in which our alternative algorithm would be ideal. For now we conclude the alternative algorithm is best suited

for sampling from highly monotone distributions where a reasonable selection of k still spans nearly the entire sample space.

6.2 Further Research

The next logical step in the exploration of potential applications of this new algorithm is to apply it to a real statistical problem. While illustrative, the simulation study above was performed on a trivial problem. It remains to be shown that the alternative algorithm can produce results comparable to or better than the Metropolis-Hastings in sampling from a distribution we actually cannot directly sample from before any major conclusions can be drawn.

A more formal argument regarding the asymptotic autocorrelation of the alternative algorithm as k approaches infinity would be another endeavor to undertake. We hope to pursue both of these questions in future works.

7. Conclusion

This report compared two MCMC methods for sampling unnormalized, discrete distributions: the Metropolis-Hastings and an alternative algorithm. The two algorithms were evaluated against one another through a two-part simulation study. The first component of the simulation study looked at their apparent convergence speed by generating multiple, independent chains. Here the alternative appeared to converge faster when using a large spread parameter and targeting a monotone distribution, while both algorithms fared similarly when a targeting less monotone distribution. The second component of the simulation study looked at the autocorrelation of samples from each algorithm. This suggested that the alternative algorithm is capable of generating samples with significantly less autocorrelation than the Metropolis-Hastings as the scale parameter increases. This result is reasonably explained by the structure of the alternative algorithm but remains to be stated formally. That withstanding, the findings here suggest this alternative algorithm shows promise as an addition to our current arsenal of MCMC methods and merits further investigation.

References

Bernardo, J., & Smith, A. (1994). *Bayesian theory.* Chichester, Eng.: Wiley.

Box, G., & Jenkins, G. (1976). *Time series analysis: Forecasting and control* (Rev. ed.). San Francisco: Holden-Day.

Cowles, M., & Carlin, B. (1996). Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. *Journal of the American Statistical Association, 91*(434), 883-904.

Gamerman, D., & Lopes, H. (2006). *Markov chain Monte Carlo: Stochastic simulation for Bayesian inference* (2nd ed.). London: Chapman & Hall.

Gelfand, A., & Smith, A. (1990). Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association, 85*(410), 398-409.

Gelman, A., Roberts, G.O., & Gilks, W.,R. (1996). Efficient Metropolis Jumping Rules. In J.M. Bernardo et al. (Eds.), *Bayesian Statistics 4*, Oxford: Oxford University Press.

Geyer, C. (1992). Practical Markov Chain Monte Carlo. *Statistical Science, 7*(4), 473-483.

Hastings, W. (1970). Monte Carlo Sampling Methods Using Markov Chains And Their Applications. *Biometrika, 57*(1), 97-109.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., & Teller, E. (1953). Equation of State Calculation by Fast Computing Machines. *The Journal of Chemical Physics, 21*(6), 1087-1092.

R Development Core Team. (2010). R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from http://www.R-project.org

Tierney, L. (1994). *Markov Chains for Exploring Posterior Distributions. The Annals of Statistics, 22*(4), 1701-1728.

Walker, S. (2014). Sampling Unnormalized Probabilities: An Alternative to the Metropolis–Hastings Algorithm. *SIAM Journal on Scientific Computing,* A482-A494.